Large-Scale Nonlinear Optimization in Circuit Tuning

Andreas Wächter

IBM T.J. Watson Research Center

Department of Mathematical Sciences

`andreasw@watson.ibm.com`

NACDM 2004

Santa Fe, New Mexico

June 25, 2004

Outline

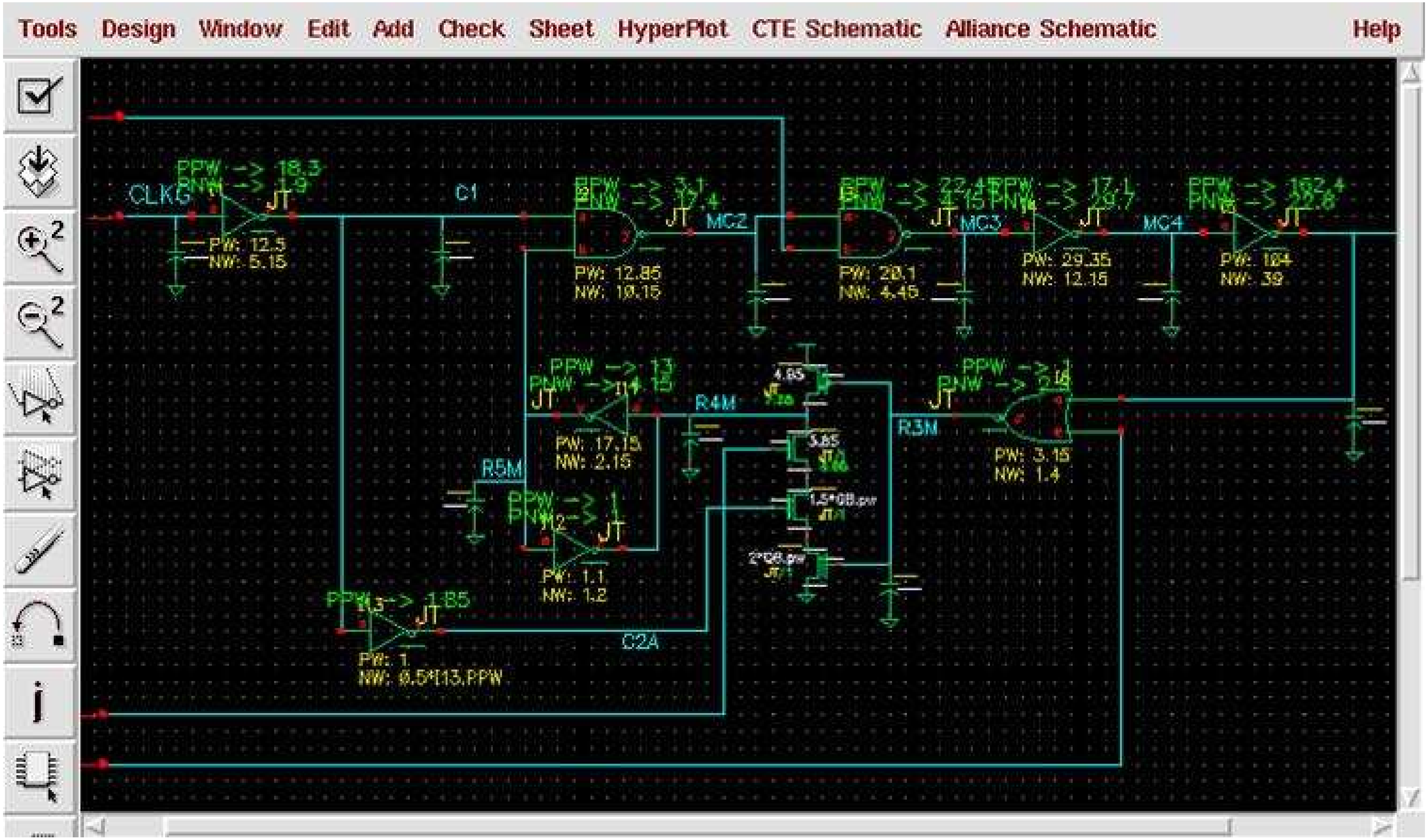
- Circuit Tuning
  - Nonlinear optimization problem formulation
  - Simulation of gates
- IPOPT
  - Interior point method for large-scale nonlinear optimization
  - Filter line search procedure
- Numerical results

Collaborators:

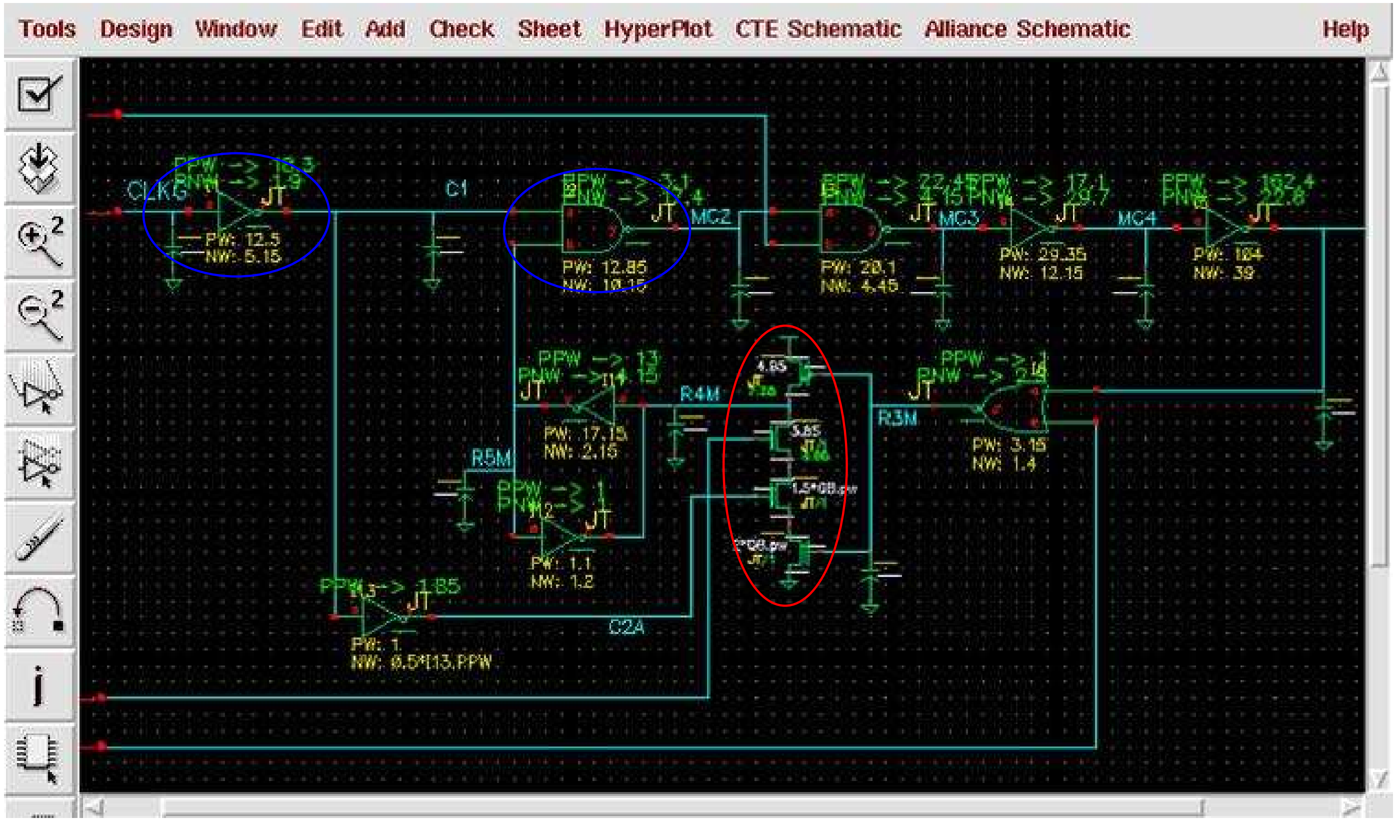
- Circuit tuning: Andrew R. Conn, Chandu Visweswariah, Michael Henderson (IBM Watson)
  EDA Department, IBM Fishkill, NY
- IPOPT: Lorenz T. Biegler (Carnegie Mellon University)

Circuit Tuning

Digital Circuit

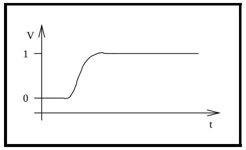


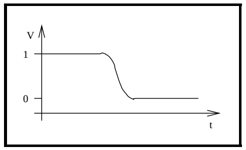
Digital Circuit

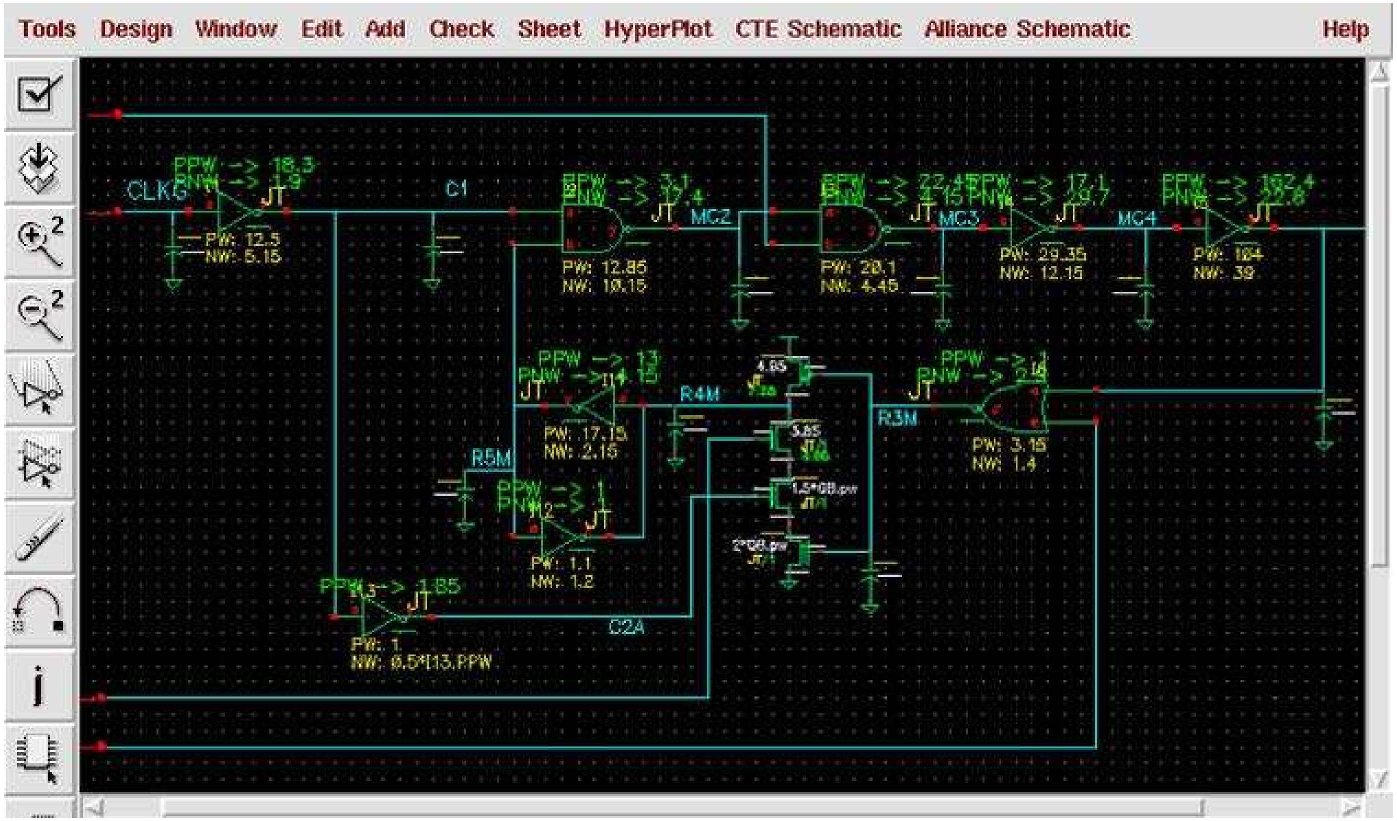


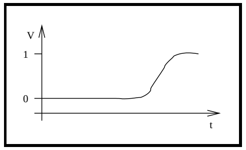
- Basic building blocks:
  - Transistors (switches); Gates (logical units)
- Connected by wires

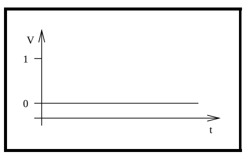
Digital Circuit

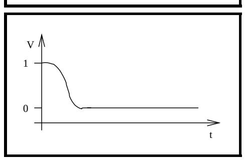






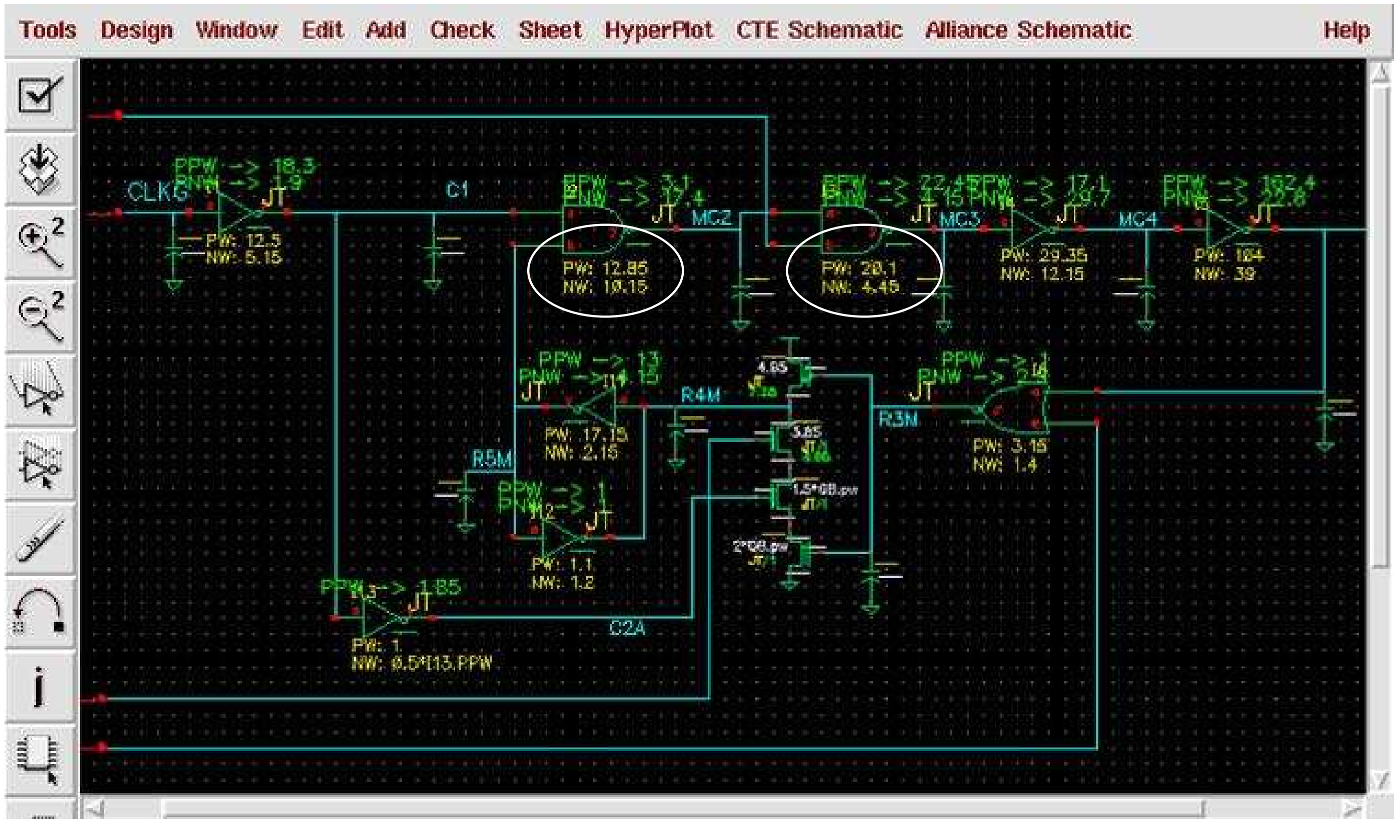






- Signals arrive at the inputs, pass through the circuit, and leave at the outputs

Digital Circuit



- Can change the “speed” of a gate by changing the widths of its transistors (PFETs and NFETs)

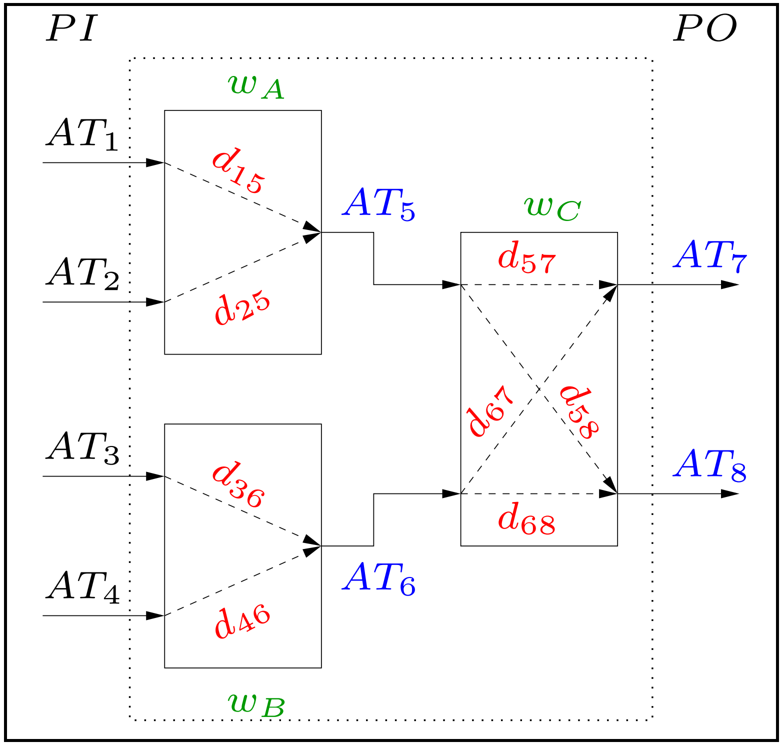
Circuit Tuning

- Want to optimize aspects of the digital circuit
  - Delay of signals
  - Area requirement
  - Power consumption
  - Combination of above

  by changing widths of transistors
- Often, overall circuit too large (CPU has few 100 million transistors)
- Split into “macros” and fix transistors $\longrightarrow$ suboptimal solutions
- Currently, we can tune circuits with up to 71,022 transistors (19,576 independent)
- Strong incentive to be able to tune larger circuits more quickly

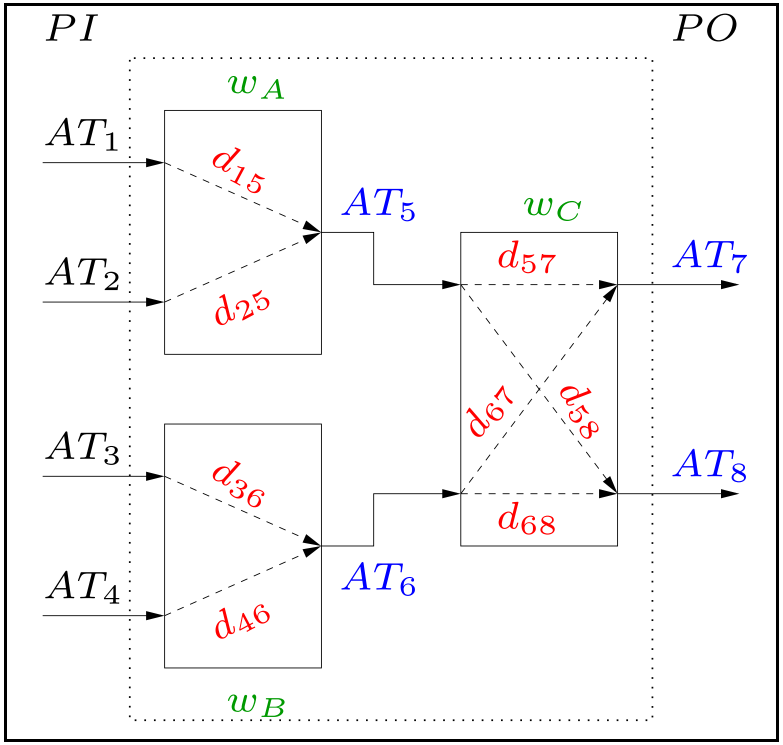
Mathematical Formulation (Static Timing)

Circuit



Mathematical Formulation (Static Timing)

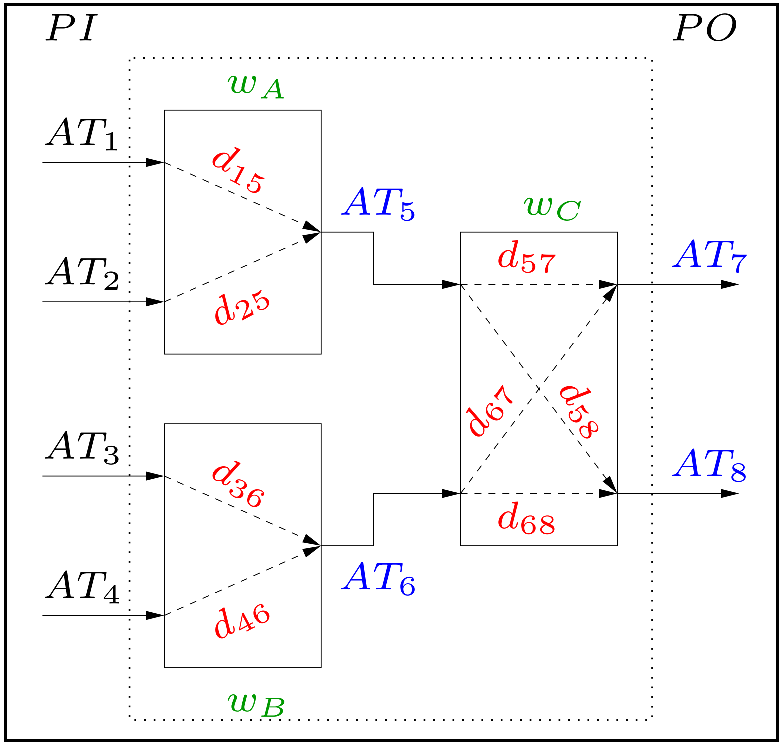
Circuit



$$
\begin{aligned}
AT_5 &= \max\{AT_1 + d_{15}, AT_2 + d_{25}\} \\
AT_6 &= \max\{AT_3 + d_{36}, AT_4 + d_{46}\} \\
AT_7 &= \max\{AT_5 + d_{57}, AT_6 + d_{67}\} \\
AT_8 &= \max\{AT_5 + d_{58}, AT_6 + d_{68}\}
\end{aligned}
$$

Mathematical Formulation (Static Timing)

Circuit

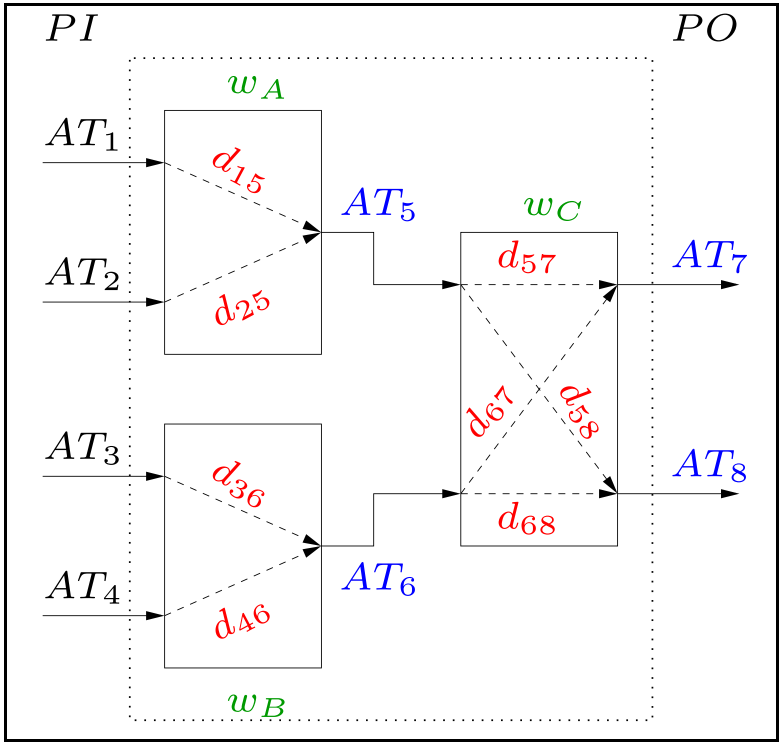


$$
\begin{aligned}
AT_5 &= \max\{AT_1 + d_{15}, AT_2 + d_{25}\} \\
AT_6 &= \max\{AT_3 + d_{36}, AT_4 + d_{46}\} \\
AT_7 &= \max\{AT_5 + d_{57}, AT_6 + d_{67}\} \\
AT_8 &= \max\{AT_5 + d_{58}, AT_6 + d_{68}\}
\end{aligned}
$$

$$AT_j = \max\{AT_i + d_{ij} : i \in \text{input}(j)\}$$

Mathematical Formulation (Static Timing)

Circuit



$$\begin{aligned} AT_5 &= \max\{AT_1 + d_{15}, AT_2 + d_{25}\} \\ AT_6 &= \max\{AT_3 + d_{36}, AT_4 + d_{46}\} \\ AT_7 &= \max\{AT_5 + d_{57}, AT_6 + d_{67}\} \\ AT_8 &= \max\{AT_5 + d_{58}, AT_6 + d_{68}\} \end{aligned}$$

$$AT_j = \max\{AT_i + d_{ij} : i \in \text{input}(j)\}$$

Delays are functions of transistor widths:

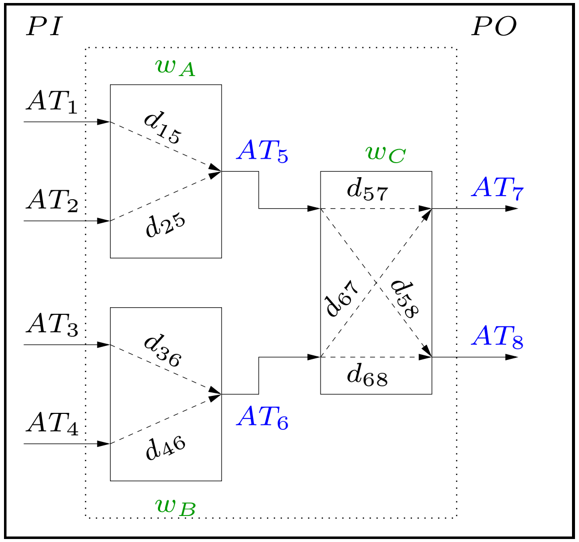
$$d_{ij} = d_{ij}(w_j, w_{\text{next}(j)})$$

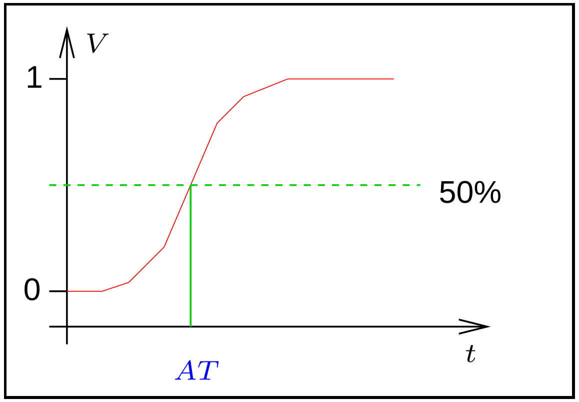
Optimization Problem (Delay Minimization)

$$\min_{AT,w} \quad \max\{AT_i \ : \ i \in PO\}$$

$$s.t. \quad AT_j = \max\{AT_i + d_{ij}(w_j, w_{\text{next}(j)}) \ : \ i \in \text{input}(j)\}$$

$$w_{\min} \leq w \leq w_{\max}$$



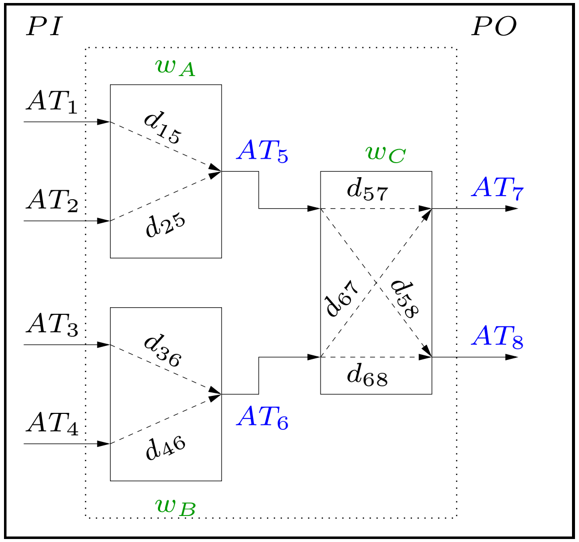


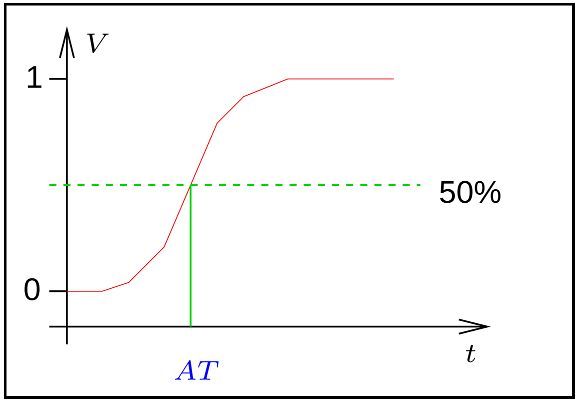
Optimization Problem (Delay Minimization)

$$\min_{AT,w} \quad \max\{AT_i \; : \; i \in PO\}$$

$$s.t. \quad AT_j = \max\{AT_i + d_{ij}(w_j, w_{\mathsf{next}(j)}) \; : \; i \in \mathsf{input}(j)\}$$

$$w_{\mathsf{min}} \leq w \leq w_{\mathsf{max}}$$





$$\min_{x} \quad \max_{i} f_i(x) \quad \longrightarrow \quad \min_{x,z} \; z \quad s.t. \; z \geq f_i(x) \quad \forall i$$

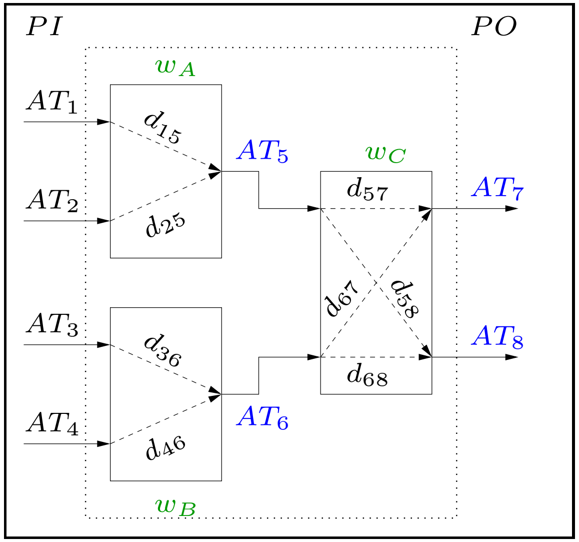
Optimization Problem (Delay Minimization)

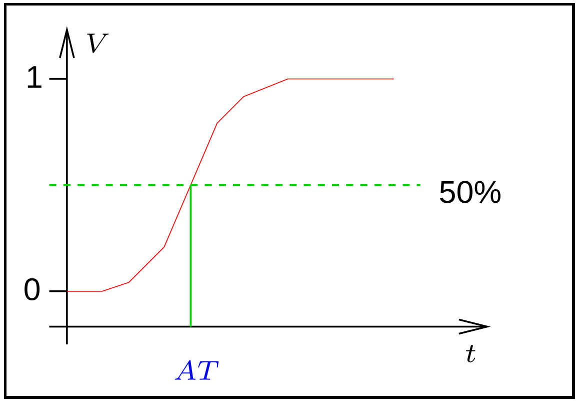
$$\min_{AT, w, z} \quad z$$

$$s.t. \quad z \geq AT_i \qquad i \in PO$$

$$AT_j \geq AT_i + d_{ij}(w_j, w_{\mathrm{next}(j)}) \qquad i \in \mathrm{input}(j)$$

$$w_{\min} \leq w \leq w_{\max}$$



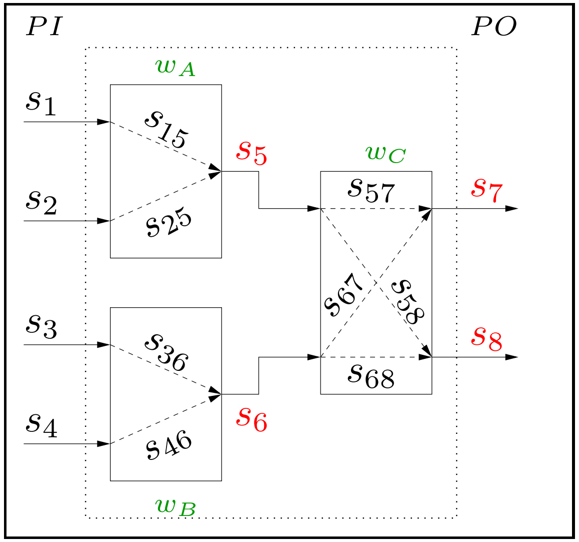


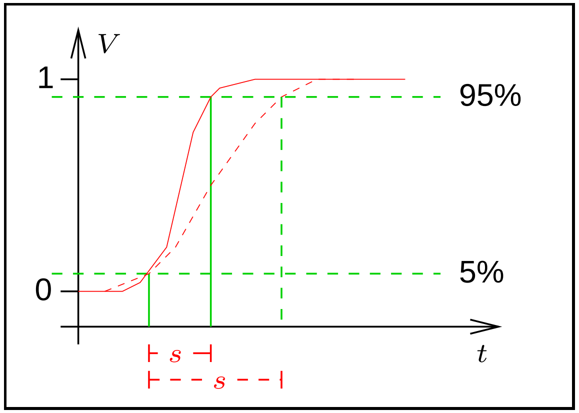
Optimization Problem (Delay Minimization)

$$\min_{AT, w, s, z} \quad z$$

$$
\begin{aligned}
s.t. \quad & z \geq AT_i & i \in PO \\
& AT_j \geq AT_i + d_{ij}(w_j, w_{\mathsf{next}(j)}, s_i) & i \in \mathsf{input}(j) \\
& s_j \geq s_{ij}(w_j, w_{\mathsf{next}(j)}, s_i) & i \in \mathsf{input}(j)
\end{aligned}
$$

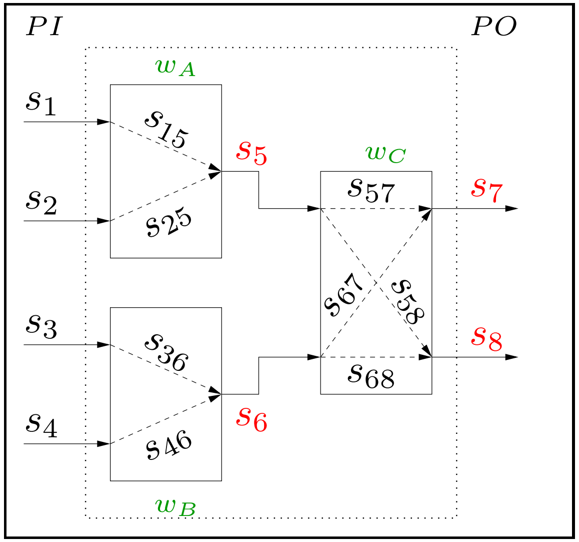
$$w_{\mathsf{min}} \leq w \leq w_{\mathsf{max}}, \qquad s_{\mathsf{min}} \leq s \leq s_{\mathsf{max}}$$

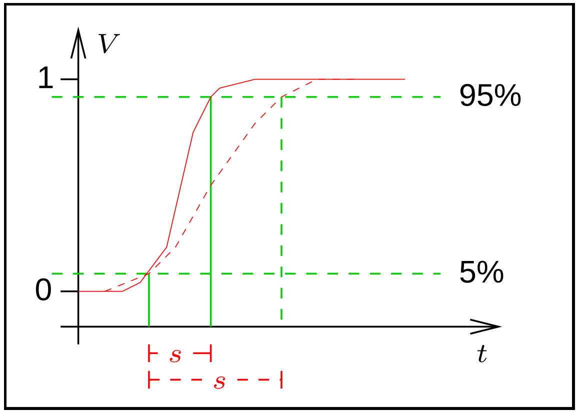




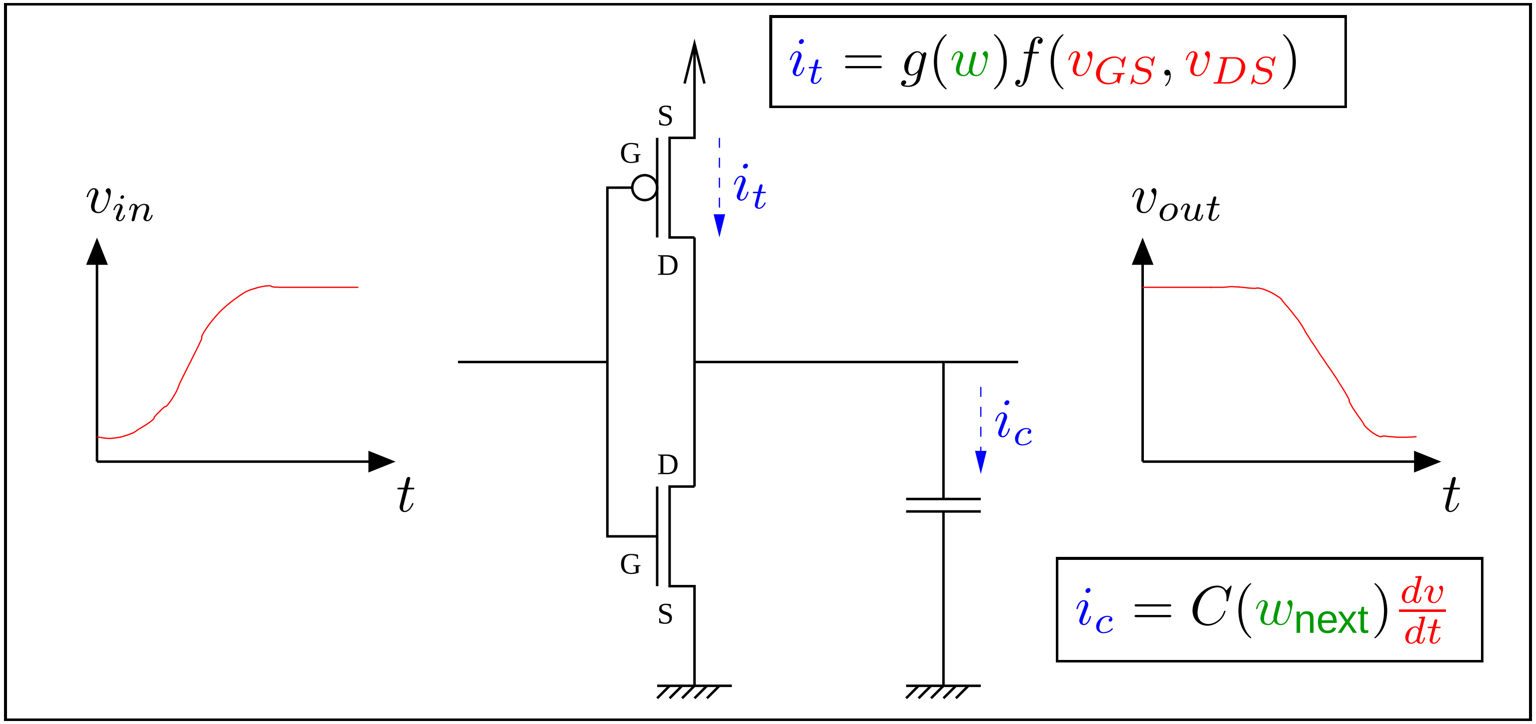
Optimization Problem (Delay Minimization)

$$
\begin{array}{lll}
\displaystyle\min_{AT,w,s,z} & z & \\
s.t. & z \geq AT_i & i \in PO \\
 & AT_j \geq AT_i + d_{ij}(w_j, w_{\text{next}(j)}, s_i) & i \in \text{input}(j) \\
 & s_j \geq s_{ij}(w_j, w_{\text{next}(j)}, s_i) & i \in \text{input}(j) \\
 & \sum A_i w_i \leq A_{\text{max}} & \\
 & C_i(w) \leq C_i^{\text{max}} & i \in PI \\
 & \beta_i^{\text{min}} \leq \dfrac{w_i^{\text{PFET}}}{w_i^{\text{NFET}}} \leq \beta_i^{\text{max}} & i \in G \\
 & w_{\text{min}} \leq w \leq w_{\text{max}}, \quad s_{\text{min}} \leq s \leq s_{\text{max}} &
\end{array}
$$





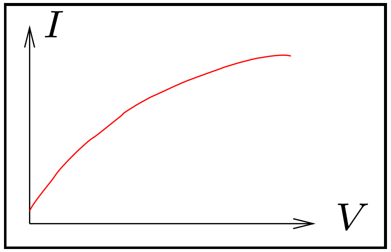
Simulation of a Gate



- System of differential and algebraic equations:
  - Conservation laws
  - (Parasitic) capacitancies: Include dv/dt terms
  - I-V characteristics for conducting devices

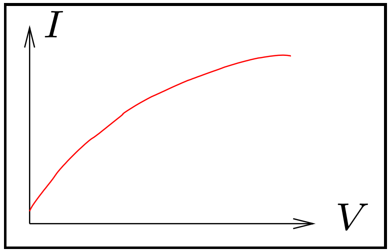
Simulation of a Gate - Numerical Methods

- "Traditional approach"
  - Solve DAE system by standard integration method
  - Step size control (in time)
  - Solve nonlinear system of equations
  - Need to evaluate I-V characteristic functions and its gradients (expensive)
  - Sensitivities expensive

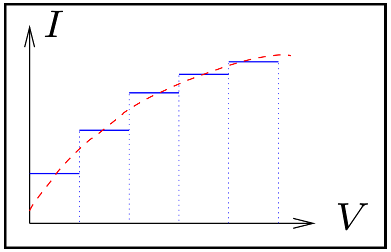


Simulation of a Gate - Numerical Methods

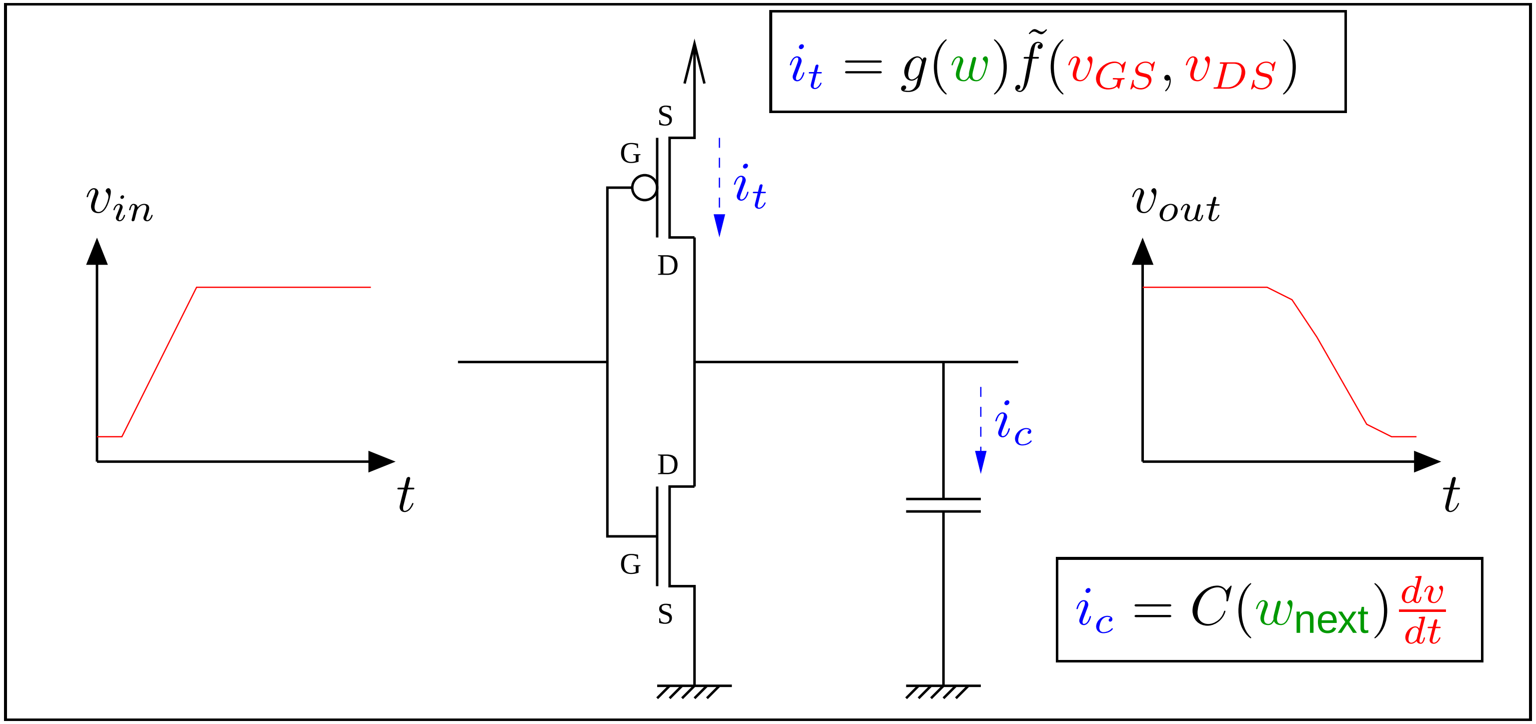
- “Traditional approach”
  - Solve DAE system by standard integration method
  - Step size control (in time)
  - Solve nonlinear system of equations
  - Need to evaluate I-V characteristic functions and its gradients (expensive)
  - Sensitivities expensive



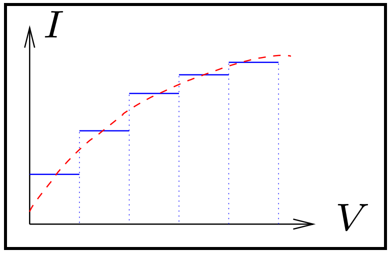
- SPECS (Visweswariah, 1991)
  - Replace I-V characteristics by piecewise constant approximations
  - Cheap table lookups
  - i piecewise constant
  - v piecewise linear
  - Very simplified numerics



Simulation with SPECS



- Keep track of i, v, and dv/dt at all nodes
- Can easily find next event time t_{event} when next segment in I-V characteristic is reached
- Update all i, v, and dv/dt in neighboring nodes



SPECS

- “Event-driven” simulator (discretize in i instead of time)
- Very fast (“local updates”, simple algebraic operations)
- Derivatives (in direct or adjoint approach) computed with little overhead
- Up to 5% timing inaccuracy
- Circuits with several 100,000 transistors simulated
- Here, only small circuits (gates) are simulated

Properties of Optimization Problem

- The nonlinear functions d_{ij} and s_{ij} are computed by simulation
  - Computationally expensive
  - First derivatives available
  - Numerical noise (from simulation)
- Many variables and many degrees of freedom
- After optimization, transistor widths are snapped to grid
  - Do not need highly accurate solution

Properties of Optimization Problem

- The nonlinear functions d_{ij} and s_{ij} are computed by simulation
  - Computationally expensive
  - First derivatives available
  - Numerical noise (from simulation)
- Many variables and many degrees of freedom
- After optimization, transistor widths are snapped to grid
  - Do not need highly accurate solution
- Alternative approach: Dynamic Tuning
  - Simulate entire circuit at once
  - Need to be given “input sequence”
  - more flexible; less pessimistic ($+$)
  - Requires very good knowledge of circuit ($-$)

EinsTuner

- IBM-internal implementation
- Original optimization engine: Lancelot (Conn, Gould, Toint)
- Lancelot had to be customized (handle noise; made aggressive)
- Preprocessing (Pruning)

EinsTuner

- IBM-internal implementation
- Original optimization engine: Lancelot (Conn, Gould, Toint)
- Lancelot had to be customized (handle noise; made aggressive)
- Preprocessing (Pruning)

- Used for the design of every custom digital chip in IBM
  - 15% gain in speed over carefully hand-tuned circuits
  - Designers can now concentrate on other (non-tuning) tasks

EinsTuner

- IBM-internal implementation
- Original optimization engine: Lancelot (Conn, Gould, Toint)
- Lancelot had to be customized (handle noise; made aggressive)
- Preprocessing (Pruning)

- Used for the design of every custom digital chip in IBM
  - 15% gain in speed over carefully hand-tuned circuits
  - Designers can now concentrate on other (non-tuning) tasks

- New optimization engine: IPOPT

IPOPT

Problem Statement

$$\boxed{\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ s.t. \quad & c(x) = 0 \\ & x_L \le x \le x_U \end{aligned}}$$

x	Variables
$f(x) : \mathbb{R}^n \longrightarrow \mathbb{R}$	Objective function
$c(x) : \mathbb{R}^n \longrightarrow \mathbb{R}^m$	Equality constraints
$x_L \in (\mathbb{R} \cup \{-\infty\})^n$	Lower bounds
$x_U \in (\mathbb{R} \cup \{\infty\})^n$	Upper bounds

- Functions $f(x)$ and $c(x)$ sufficiently smooth (usually C^2)
- General inequality constraints

$$d(x) \le 0$$

can be reformulated as

$$d(x) + s = 0, \quad s \ge 0$$

Barrier Methods

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \quad & f(x) \\ s.t. \quad & c(x) = 0 \\ & x \geq 0 \end{aligned}$$

Barrier Methods

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ s.t. \quad & c(x) = 0 \\ & x \geq 0 \end{aligned}$$

$$\downarrow$$

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) - \mu \sum_{i=1}^{n} \ln(x^{(i)}) \\ s.t. \quad & c(x) = 0 \end{aligned}$$

Barrier Parameter: $\mu > 0$

Idea: $x_*(\mu) \to x_*$ as $\mu \to 0$.

- Solve a sequence of barrier problems to increasingly tighter tolerances
  - Fiacco, McCormick (1968)

Barrier Methods

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) \\ s.t. \quad & c(x) = 0 \\ & x \geq 0 \end{aligned}$$

$$\downarrow$$

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & f(x) - \mu \sum_{i=1}^{n} \ln(x^{(i)}) \\ s.t. \quad & c(x) = 0 \end{aligned}$$

Barrier Parameter: $\mu > 0$
Idea: $x_*(\mu) \to x_*$ as $\mu \to 0$.

- Solve a sequence of barrier problems to increasingly tighter tolerances
  - Fiacco, McCormick (1968)
- Interior point NLP algorithms
  - El-Bakry, Tapia, Tsuchiya, Zhang (1996)
  - Benson, Shanno, Vanderbei (1997/2003) [LOQO]
  - Yamashita (1998)
  - Forsgren, Gill (1998)
  - Byrd, Gilbert, Hribar, Nocedal, Waltz (1999/2003) [KNITRO]
  - W, Biegler (1999/2004) [IPOPT]
  - Ulbrich, Ulbrich, Vicente (2000)
  - Gould, Orban, Toint (2003) [SUPERB]
  - etc.

Computation of Search Direction

Barrier Problem (fixed μ)

$$\min_{x \in \mathbb{R}^n} \quad \varphi_\mu(x) := f(x) - \mu \sum \ln(x^{(i)})$$
$$s.t. \quad c(x) = 0$$

Computation of Search Direction

Barrier Problem (fixed μ)

$$\min_{x \in \mathbb{R}^n} \quad \varphi_\mu(x) := f(x) - \mu \sum \ln(x^{(i)})$$
$$s.t. \quad c(x) = 0$$

Optimality Conditions

$$\begin{aligned} \nabla\varphi_\mu(x) + \nabla c(x)\lambda &= 0 \\ c(x) &= 0 \\ (x &> 0) \end{aligned}$$

Computation of Search Direction

Barrier Problem (fixed μ)

$$\min_{x \in \mathbb{R}^n} \quad \varphi_\mu(x) := f(x) - \mu \sum \ln(x^{(i)})$$
$$s.t. \quad c(x) = 0$$

Optimality Conditions

$$\begin{aligned} \nabla \varphi_\mu(x) + \nabla c(x)\lambda &= 0 \\ c(x) &= 0 \\ (x &> 0) \end{aligned}$$

Apply Newton's Method

$$\begin{bmatrix} W_k & \nabla c(x_k) \\ \nabla c(x_k)^T & 0 \end{bmatrix} \begin{pmatrix} \Delta x_k \\ \Delta \lambda_k \end{pmatrix} = - \begin{pmatrix} \nabla \varphi_\mu(x_k) + \nabla c(x_k)\lambda_k \\ c(x_k) \end{pmatrix}$$

Here:

- $W_k \approx \nabla^2_{xx} \mathcal{L}_\mu(x_k, \lambda_k)$
- $\mathcal{L}_\mu(x, \lambda) = \varphi_\mu(x) + c(x)^T \lambda$

Computation of Search Direction

Barrier Problem (fixed μ)

$$\min_{x\in\mathbb{R}^n} \quad \varphi_\mu(x) := f(x) - \mu \sum \ln(x^{(i)})$$
$$s.t. \quad c(x) = 0$$

Optimality Conditions

$$\begin{aligned} \nabla\varphi_\mu(x) + \nabla c(x)\lambda &= 0 \\ c(x) &= 0 \\ (x &> 0) \end{aligned}$$

Apply Newton's Method

$$\begin{bmatrix} W_k & \nabla c(x_k) \\ \nabla c(x_k)^T & 0 \end{bmatrix} \begin{pmatrix} \Delta x_k \\ \Delta\lambda_k \end{pmatrix} = -\begin{pmatrix} \nabla\varphi_\mu(x_k) + \nabla c(x_k)\lambda_k \\ c(x_k) \end{pmatrix}$$

Here:

- $W_k \approx \nabla^2_{xx}\mathcal{L}_\mu(x_k, \lambda_k)$
- $\mathcal{L}_\mu(x, \lambda) = \varphi_\mu(x) + c(x)^T\lambda$

- Use primal-dual approach
- Matrix becomes very ill-conditioned
- Need to ensure descent properties

Line Search

Need to find $\alpha_k \in (0, 1]$ to obtain new iterates

$$\boxed{(x_{k+1}, \lambda_{k+1}) \quad = \quad (x_k, \lambda_k) + \alpha_k\, (\Delta x_k, \Delta\lambda_k)}$$

Line Search

Need to find $\alpha_k \in (0,1]$ to obtain new iterates

$$\boxed{(x_{k+1}, \lambda_{k+1}) \quad = \quad (x_k, \lambda_k) + \alpha_k \, (\Delta x_k, \Delta \lambda_k)}$$

1. Keep x_k positive (“fraction-to-the-boundary rule”):
   Determine largest $\alpha_k^\tau \in (0,1]$ such that ($\tau \approx 0.99$)

$$\boxed{x_k + \alpha_k^\tau \Delta x_k \quad \geq \quad (1-\tau) x_k \quad > \quad 0}$$

Line Search

Need to find $\alpha_k \in (0,1]$ to obtain new iterates

$$\boxed{(x_{k+1}, \lambda_{k+1}) \quad = \quad (x_k, \lambda_k) + \alpha_k\,(\Delta x_k, \Delta\lambda_k)}$$

1. Keep x_k positive (“fraction-to-the-boundary rule”):
   Determine largest $\alpha_k^\tau \in (0,1]$ such that ($\tau \approx 0.99$)

   $$\boxed{x_k + \alpha_k^\tau \Delta x_k \quad \geq \quad (1-\tau)x_k \quad > \quad 0}$$

2. Backtracking line search with $\alpha_k = \alpha_k^\tau, \frac{1}{2}\alpha_k^\tau, \frac{1}{4}\alpha_k^\tau, \ldots$ to ensure global convergence (to first-order optimal point)
   - Line search filter method

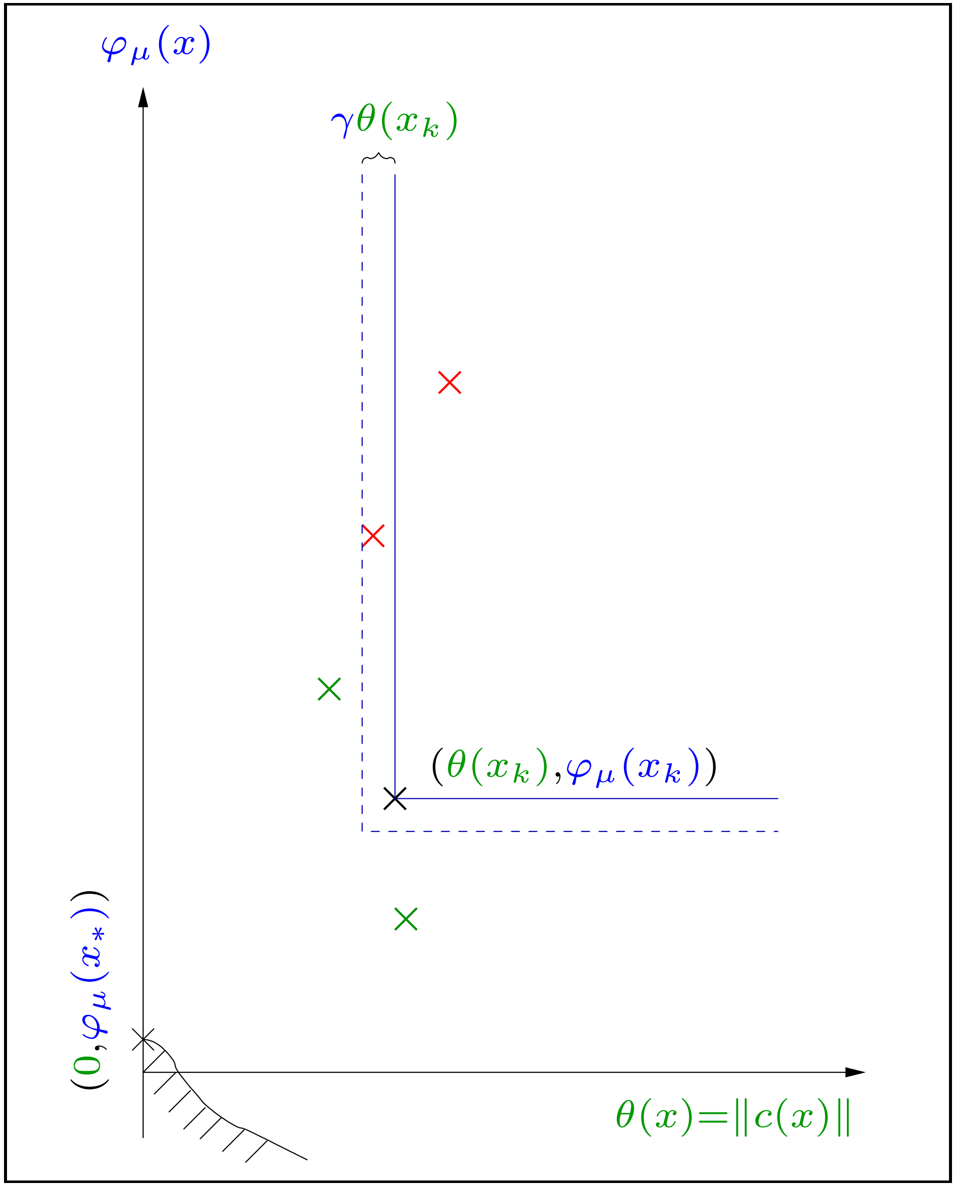
A Line Search Filter Method

- Fletcher, Leyffer (1998), . . .
- Alternative to merit functions

Idea:

$$\boxed{\begin{array}{ll} \min & \varphi_\mu(x) \\ s.t. & c(x) = 0 \end{array}}$$

$$\boxed{\min \quad \theta(x)} \qquad \boxed{\min \quad \varphi_\mu(x)}$$

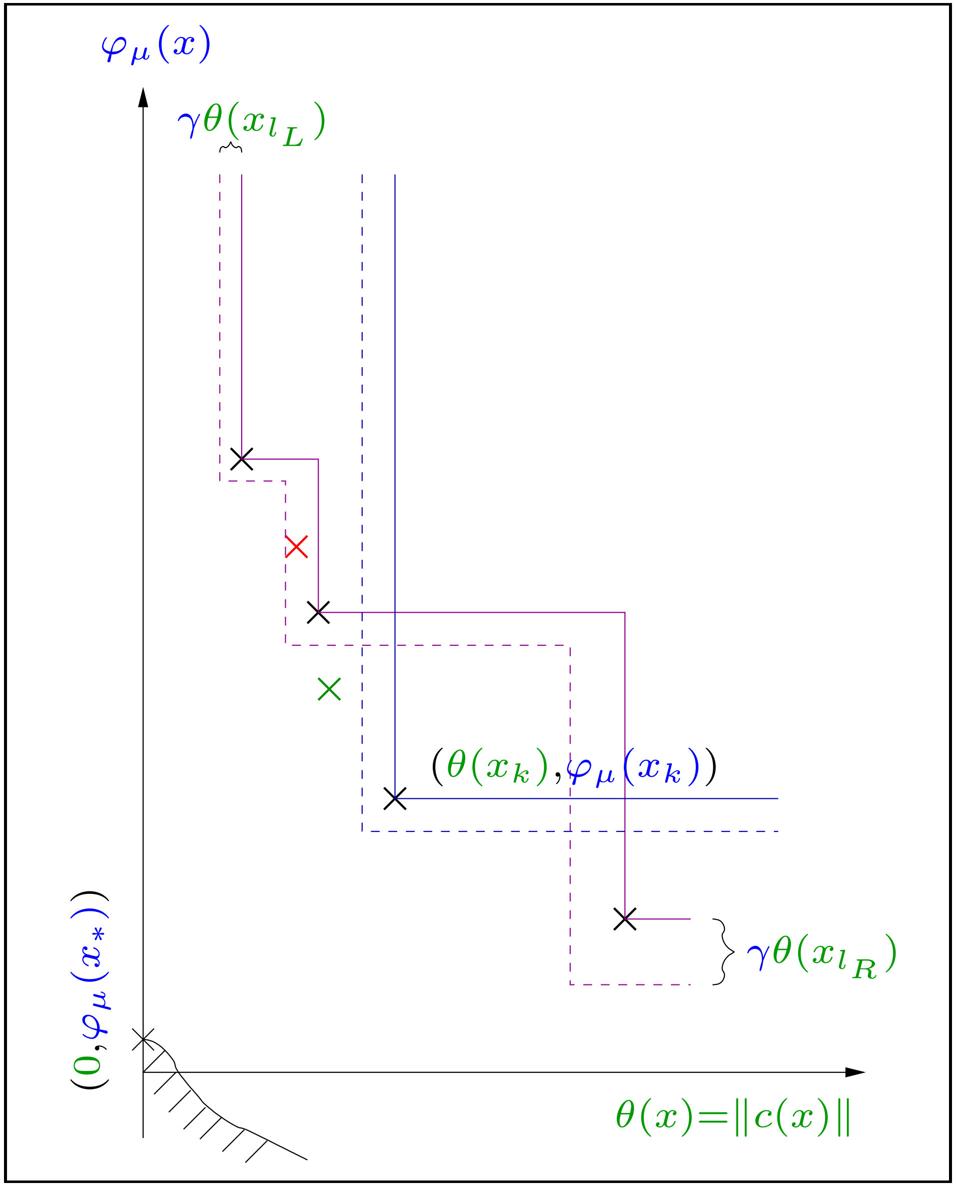


A Line Search Filter Method

Need to avoid cycling

$\Downarrow$

Store some previous
$(\theta(x_l), \varphi_\mu(x_l))$ pairs in *Filter*



IPOPT

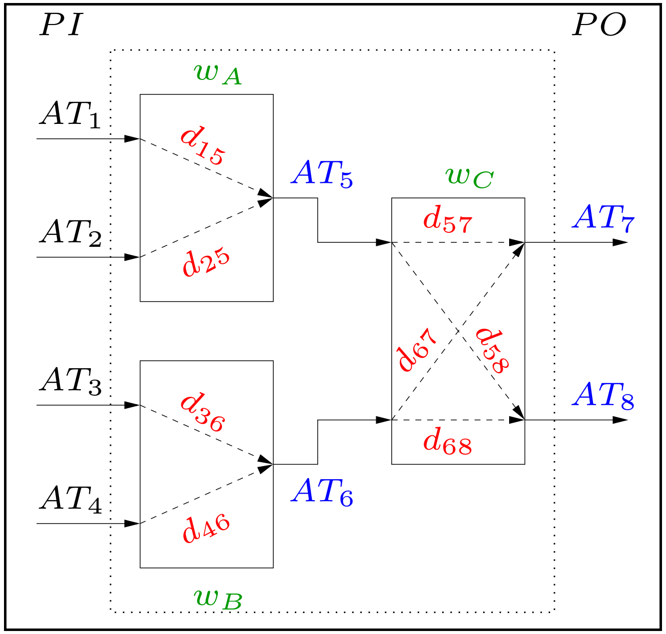
- Implemented as IPOPT (Fortran 77 / C)   [soon C++]
- Compares well with other NLP solvers as general purpose code
- Available as open source from COIN-OR

  `http://www.coin-or.org/Ipopt`

- Includes interfaces to AMPL and CUTEr/SIF   [soon Matlab, GAMS]
- Is available at Argonne's NEOS Server
- Used for
  - Dynamic optimization (discretized DAE constraints)
  - Nonlinear model predictive control
  - Parameter estimation
  - MPCC (Raghunathan, Biegler, 2004)

Circuit Tuning + IPOPT

Integration of IPOPT in EinsTuner



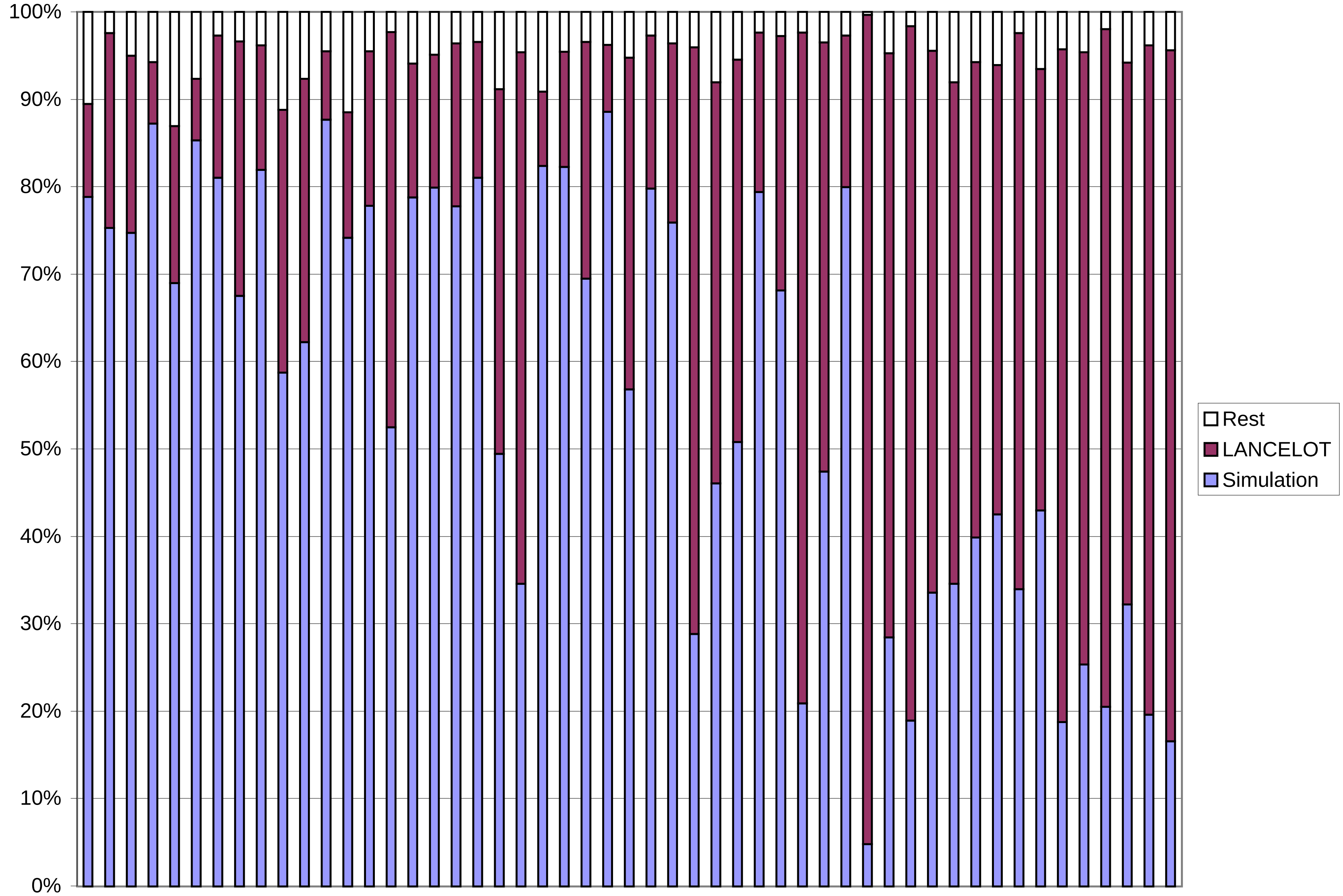
$$
\begin{aligned}
\min_{AT,w} \quad & AT_{\text{latest}} \\
s.t. \quad & AT_{\text{latest}} \geq AT_i \\
& AT_j \geq AT_i + d_{ij}(w_i) \\
& \dots \\
& w_{\text{min}} \leq w \leq w_{\text{max}}
\end{aligned}
$$

- Approximate 2nd derivatives with limited-memory BFGS
- Factorize linear system with WSMP (Gupta), a sparse direct parallel solver
- Overall faster and more robust than previous optimization engine
- Released into production

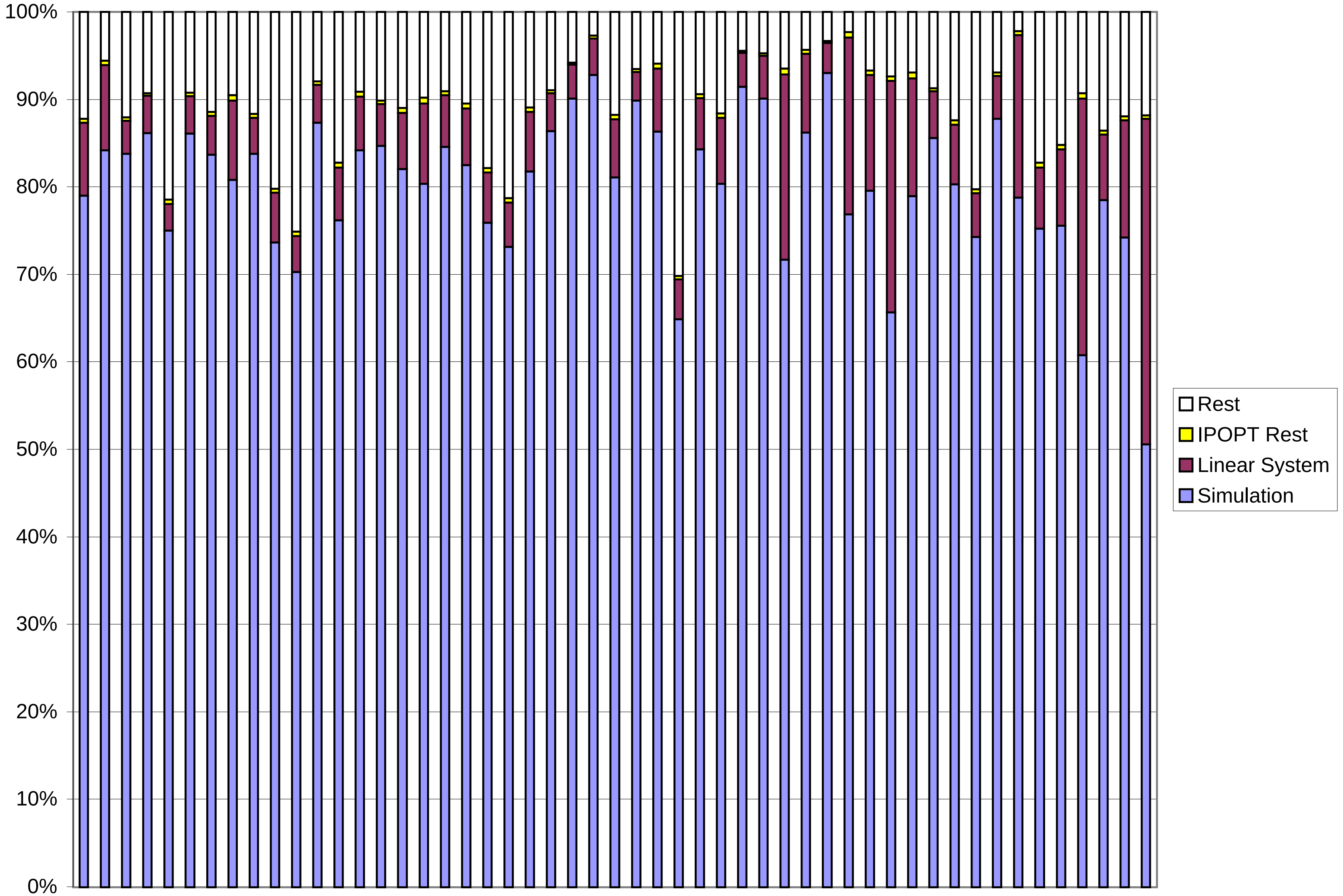
Next slide:

- 51 benchmark problems (n=1,261,. . . ,161,701)

Breakdown of CPU time — Lancelot



Breakdown of CPU time — Ipopt



Conclusion

- Circuit Tuning
  - Large-scale nonlinear optimization problem
  - Gate simulation by event-driven simulator SPECS
  - Used for the design of every custom digital circuit at IBM
- IPOPT
  - Barrier method
  - Line search filter method
  - Good practical performance as general purpose NLP solver
  - Increased performance in EinsTuner and allows parallel version

`http://www.coin-or.org/Ipopt`

References

http://www.research.ibm.com/people/a/andreasw

- A. Wächter, C. Visweswariah and A. R. Conn (2003)
  Large-Scale Nonlinear Optimization in Circuit Tuning
  to appear in: Future Generation Computer Systems, special issue on the Speedup/PARS workshop in Basel, Switzerland
- A. Wächter and L. T. Biegler (March 2004)
  On the Implementation of an Interior-Point Filter Line-Search Algorithm for Large-Scale Nonlinear Programming
  Research Report RC 23149, IBM T. J. Watson Research Center, Yorktown, USA
- A. Wächter and L. T. Biegler (August 2001, revised Februrary 2004)
  Line Search Filter Methods for Nonlinear Programming: Motivation and Global Convergence
  Research Report RC 23036, IBM T. J. Watson Research Center, Yorktown, USA
- A. Wächter and L. T. Biegler (August 2001, revised Februrary 2004)
  Line Search Filter Methods for Nonlinear Programming: Local Convergence
  Research Report RC 23033, IBM T. J. Watson Research Center, Yorktown, USA